CLAIM AMENDMENTS

This listing of claims will replace all prior versions, and listings, of claims in the application:

1. – 8. (Cancelled)

9. (Currently Amended) A machine-readable medium that provides executable firmware instructions which, when executed by a processor in a computer system having a native environment that executes in physical mode, cause the processor to perform operations comprising:

implementing an extensible firmware framework via which firmware modules are loaded during a pre-boot phase of the computer system;

implementing a firmware-based virtual machine monitor (VMM) upon the computing system;

emulating legacy hardware components that are not present in the native environment using the VMM to provide support for legacy code running on the computer system; [[and]]

restricting access by the firmware modules to a subset of system resources provided by the native environment and to a subset of a memory map of the native environment, via the VMM; and

authenticating, via the VMM, at least one of the firmware modules that is loaded during the pre-boot phase by comparing a digital signature provided with the at least one of the firmware modules with valid digital signatures stored in a secure storage that is accessible to the VMM, but which the VMM makes inaccessible to the firmware modules and the legacy code.

10. (Cancelled)

11. (Previously Presented) The machine-readable medium of claim 9, wherein the VMM provides at least one of PC/AT hardware emulation and PC/AT environment emulation.

12. – 14. (Cancelled)

15. (Currently Amended) An apparatus comprising:

a computing system having a native execution environment that executes in physical mode, the computer system including an extensible firmware framework via which firmware modules are loaded during a pre-boot phase of the computer system; and

a virtual machine monitor ("VMM") implemented thereon, the ~~virtual machine monitor~~ VMM emulating legacy hardware components that are not present in the native environment to provide support for legacy code to run on the computer system, the VMM restricting access by the firmware modules to a subset of system resources provided by the native environment and to a subset of a memory map of the native environment, the ~~virtual machine monitor~~ VMM further authenticating [[a]]the firmware modules loaded during the pre-boot phase by comparing [[a]] digital signatures provided with the firmware modules with valid digital signatures stored in secure storage accessible to the VMM, but which the VMM makes inaccessible to the firmware modules.

16. (Cancelled)

17. (Previously Presented) The apparatus of claim 15, wherein the VMM provides at least one of PC/AT hardware emulation and PC/AT environment emulation.

18. – 26. (Cancelled)

27. (Currently Amended) A method, comprising:

implementing a virtual machine monitor (VMM) during the pre-boot phase of a computer system;

storing digital signatures of valid firmware modules in secure storage accessible to the VMM, but which the VMM makes inaccessible to other code running on the computer system; and

authenticating a firmware module via the VMM by comparing a digital signature provided with the firmware module to the digital signatures in the secure storage.

28. (Cancelled)

29. (Previously Presented) The method of claim 27, further comprising:

maintaining an attestation log via the VMM identifying firmware modules that have been loaded and authenticated by the VMM.

30. – 36. (Cancelled)

37. (Currently Amended) The apparatus of claim 15, wherein the VMM performs further operations, including:

enabling a legacy option ROM (read-only memory) to run and effect its input/output (I/O) services; [[and]]

trapping calls to legacy interrupts by the VMM; and

mapping the calls to legacy interrupts to corresponding native environment interrupts by the VMM.

~~translating the results of the I/O services into a native API (application program interface).~~

38. (New) The machine-readable medium of claim 9, wherein execution of the instructions cause further operations to be performed, comprising:

unloading one of the firmware modules that attempts to access a restricted one of the system resources or a restricted portion of the memory map.

39. (New) The machine-readable medium of claim 9, wherein authenticating, via the VMM, the at least one of the firmware modules that is loaded during the pre-boot phase further comprises:

calling signature logic of the VMM;

comparing the digital signature provided with the at least one of the firmware modules against the valid digital signatures stored in the secure storage by the signature logic to authenticate the at least one of the firmware modules; and

loading the at least one of the firmware modules, only if the at least one of the firmware modules is authenticated by the signature logic.

40. (New) The machine-readable medium of claim 39, wherein authenticating, via the VMM, the at least one of the firmware modules that is loaded during the pre-boot phase further comprises:

maintaining an attestation log via the VMM identifying firmware modules that have been loaded and authenticated by the signature logic.